

①

Name :- Jaskirat Singh

Class :- CSE-1, 8th Semester

Roll No :- 05413202717

Subject :- Machine Learning External Practical

Subject Code :- ETCS-454

Aim: Devise a ML method for your incoming mails

Theory % Algorithms %

1. In the first step feature engineering, we focus on extracting features of text. We need numerical features as input for our classifier.
2. In the non-naive Bayes way, we look at sentences in entirety, thus once the sentence does not show up in the training set, we will get a zero probability, making it difficult for further calculations.
3. In the final step we are good to go: Simply calculating the probabilities and compare which has higher probability.

(2)

Naive Bayes classifiers are a collection of Classification Algorithms based on Bayes theorem.

It is not a Single Algorithm but a family of algorithms where all of them share a common principle i.e.

every feature being classified is independent of each other.

Procedure:

For completion we would follow these steps:

- ① Loading data set :- we first load the dataset that we have used i.e. loading .csv file
- ② Preprocessing :- ~~For~~ For preparing the raw data and making it suitable for a machine learning model data processing is done
- ③ Visualization :- For the representation of data or information in a graph, chart or other visual format this step is performed.

Code: ① Loading Dataset

```
import pandas as pd;
```

```
df = pd.read_csv("spam-ham-dataset.csv")
```

~~df.head()~~

3

~~df.tail()~~

Preprocessing :

df.head()

df.tail()

df.info()

df.shape()

df.columns.values

df.columns

Visualization :

df['label-num'].value_counts()

import matplotlib.pyplot as plt

import seaborn as sns

sns.countplot(df['label-num'])

from sklearn.feature_extraction.text import
CountVectorizer

vector = CountVectorizer()

spammham = vector.fit_transform(df['text'])

spammham.toarray

4

x = spam-ham

y = df['label-ham'].values

```
from sklearn.model_selection import train_test_split
x_train, y_train, x_test, y_test = train_test_split(x, y,
                                                    test_size = 0.2, random_state
                                                    = 42)
```

```
from sklearn.naive_bayes import MultinomialNB
```

```
nb = MultinomialNB()
```

```
nb.fit(x_train, y_train)
```

```
y_pred = nb.predict(x_train)
```

```
y_pred
```

```
y_pred_test = nb.predict(x_test)
```

```
y_pred_test
```

```
from sklearn.metrics import classification_report,
ConfusionMatrix, accuracy_score
```

```
cm_test = ConfusionMatrix(y_test, y_pred_test)
```

```
cm_train = ConfusionMatrix(y_train, y_pred)
```

```
cm_test
```

```
cm_train
```

```
sns.heatmap(cm_train, annot=True)
```

Screenshots:

Reading CSV

Estimating the Relation

Visualization

Training and testing the dataset

Classification report and Confusion matrix

LOADING DATA SET:

```
In [1]: import pandas as pd
```

```
In [2]: df = pd.read_csv("spam_ham_dataset.csv")
df
```

Out[2]:

	Unnamed: 0	label	text	label_num
0	605	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
1	2349	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	3624	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	4685	spam	Subject: photoshop , windows , office . cheap ...	1
4	2030	ham	Subject: re : indian springs\r\nthis deal is t...	0
...
5166	1518	ham	Subject: put the 10 on the ft\r\nthe transport...	0
5167	404	ham	Subject: 3 / 4 / 2000 and following noms\r\nhnp...	0
5168	2933	ham	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	1409	ham	Subject: industrial worksheets for august 2000...	0
5170	4807	spam	Subject: important online banking alert\r\nindea...	1

5171 rows × 4 columns

PREPROCESSING:

```
In [3]: df.head()
```

```
Out[3]:
```

	Unnamed: 0	label	text	label_num
0	605	ham	Subject: enron methanol ; meter # : 988291\r\n...	0
1	2349	ham	Subject: hpl nom for january 9 , 2001\r\n(see...	0
2	3624	ham	Subject: neon retreat\r\nho ho ho , we ' re ar...	0
3	4685	spam	Subject: photoshop , windows , office . cheap ...	1
4	2030	ham	Subject: re : indian springs\r\nthis deal is t...	0

```
In [4]: df.tail()
```

```
Out[4]:
```

	Unnamed: 0	label	text	label_num
5166	1518	ham	Subject: put the 10 on the ft\r\nthe transport...	0
5167	404	ham	Subject: 3 / 4 / 2000 and following noms\r\nhp...	0
5168	2933	ham	Subject: calpine daily gas nomination\r\n>\r\n...	0
5169	1409	ham	Subject: industrial worksheets for august 2000...	0
5170	4807	spam	Subject: important online banking alert\r\nidea...	1

```
In [5]: df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5171 entries, 0 to 5170
Data columns (total 4 columns):
#   Column      Non-Null Count  Dtype
---  -
0   Unnamed: 0   5171 non-null   int64
1   label        5171 non-null   object
2   text         5171 non-null   object
3   label_num    5171 non-null   int64
dtypes: int64(2), object(2)
memory usage: 161.7+ KB
```

```
In [6]: df.shape
```

```
Out[6]: (5171, 4)
```

```
In [7]: df.columns.values
```

```
Out[7]: array(['Unnamed: 0', 'label', 'text', 'label_num'], dtype=object)
```

```
In [8]: df.corr()
```

```
Out[8]:
```

	Unnamed: 0	label_num
Unnamed: 0	1.000000	0.785847
label_num	0.785847	1.000000

VISUALIZATION:

```
In [10]: df['label_num'].value_counts()
```

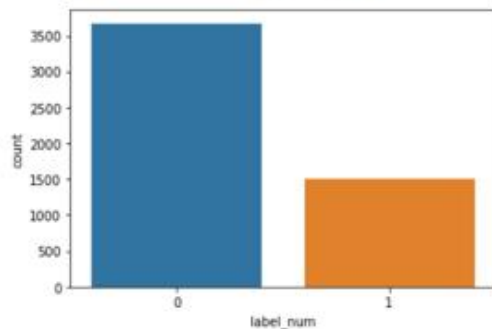
```
Out[10]: 0    3672  
        1    1499  
        Name: label_num, dtype: int64
```

```
In [11]: import matplotlib.pyplot as plt  
import seaborn as sns
```

```
In [12]: sns.countplot(df['label_num'])
```

C:\Users\is_dhillon\miniconda3\lib\site-packages\seaborn_decorators.py:36: FutureWarning: Pass the following variable as a keyword arg: x. From version 0.12, the only valid positional argument will be `data`, and passing other arguments without an explicit keyword will result in an error or misinterpretation.
warnings.warn()

```
Out[12]: <AxesSubplot:xlabel='label_num', ylabel='count'>
```



```
In [13]: from sklearn.feature_extraction.text import CountVectorizer
```

```
In [14]: vector = CountVectorizer()  
spam_ham = vector.fit_transform(df['text'])  
spam_ham.toarray
```

```
Out[14]: <bound method _cs_matrix.toarray of <5171x50447 sparse matrix of type '<class 'numpy.int64'>' with 456145 stored elements in Compressed Sparse Row format>>
```

```
In [15]: x = spam_ham  
y = df['label_num'].values  
y
```

```
Out[15]: array([0, 0, 0, ..., 0, 0, 1], dtype=int64)
```

```
In [16]: from sklearn.model_selection import train_test_split  
xtrain,xtest,ytrain,ytest = train_test_split(x,y,test_size=0.2, random_state=42)
```

```
In [17]: from sklearn.naive_bayes import MultinomialNB  
nb = MultinomialNB()  
nb.fit(xtrain,ytrain)
```

```
Out[17]: MultinomialNB()
```



```
In [18]: ypred = nb.predict(xtrain)
ypred
```

```
Out[18]: array([0, 0, 0, ..., 1, 0, 0], dtype=int64)
```

```
In [19]: ypredtest = nb.predict(xtest)
ypredtest
```

```
Out[19]: array([0, 1, 0, ..., 1, 0, 0], dtype=int64)
```

```
In [20]: from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
cmtest = confusion_matrix(ytest, ypredtest)
cmtrain = confusion_matrix(ytrain, ypred)
cmtest
```

```
Out[20]: array([[731, 11],
               [ 11, 282]], dtype=int64)
```

```
In [21]: cmtrain
```

```
Out[21]: array([[2909, 21],
               [ 33, 1173]], dtype=int64)
```

```
In [22]: sns.heatmap(cmtrain, annot=True)
```

```
Out[22]: <AxesSubplot:>
```



```
In [23]: sns.heatmap(cmtest, annot=True)
```

```
Out[23]: <AxesSubplot:>
```



```
In [24]: accuracy_score(ytest, ypredtest)
```

```
Out[24]: 0.978743961352657
```

```
In [25]: classification_report(ypredtest,ytest)
```

```
Out[25]: '
precision    recall  f1-score   support\n
0.96    0.96    0.96    293\n
0.97    1035\nweighted avg    0.98    0.98    0.98    1035\n
0.99    0.99    0.99    742\n0.98    1035\nmacro avg    0.97    0.97
```